Introduction to FPGAs and VHDL

ELEC 204 Digital Systems Design

2016

KOÇ UNIVERSITY

Standard Logic Circuits

Disadvantages:

- A lot of transistors,
- Considerable amount of circuit board space,
- A great deal of time and cost in inserting, soldering, and testing.
- No fast prototyping.

Programmable Logic Devices

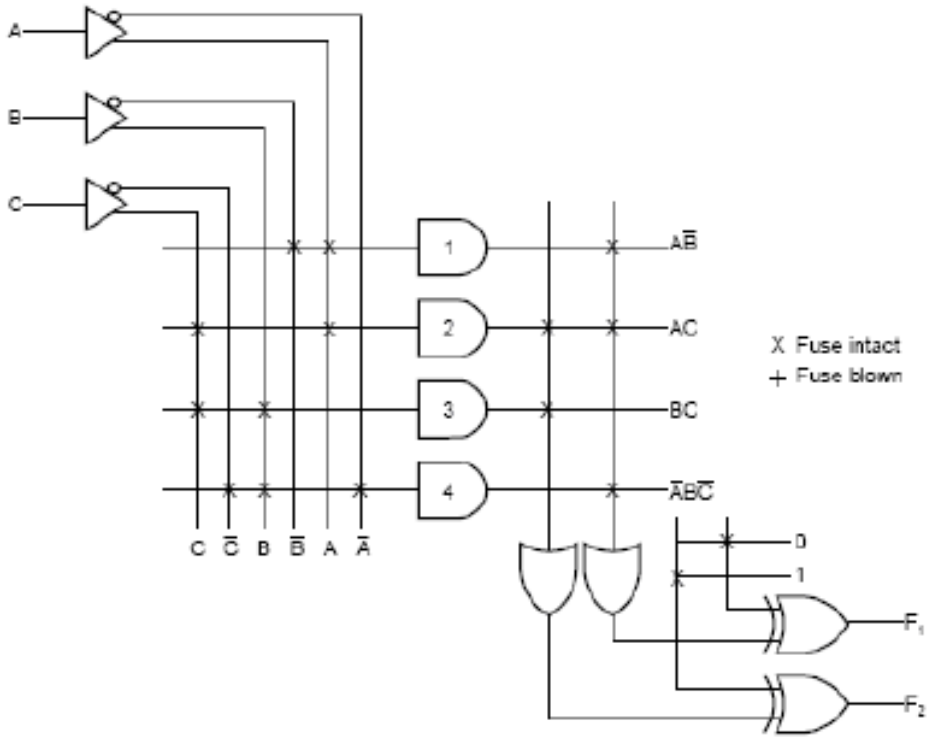
Advantages of PLDs:

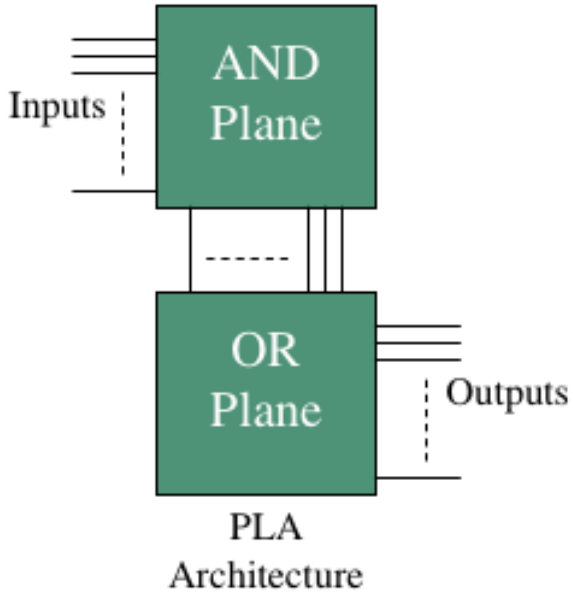
- Less board space,
- Lower power requirements,
- Higher reliability (fewer integrated circuits and circuit connections means easier troubleshooting),
- A logic design can be implemented and reconfigured by the end user,
- Rapid prototyping (if design doesn't work, fix it and reprogram the PLD).

Programmable Logic Device Types

- ***PLA*** — a Programmable Logic Array (PLA) is a relatively small FPD that contains two levels of logic, an AND-plane and an OR-plane, where both levels are programmable
- ***PAL*** — a Programmable Array Logic (PAL) is a relatively small FPD that has a programmable AND-plane followed by a fixed OR-plane
- ***SPLD*** — refers to any type of Simple PLD, usually either a PLA or PAL
- ***CPLD*** — a more Complex PLD that consists of an arrangement of multiple SPLD-like blocks on a single chip.
- ***FPGA*** — a Field-Programmable Gate Array is an FPD featuring a general structure that allows very high logic capacity.

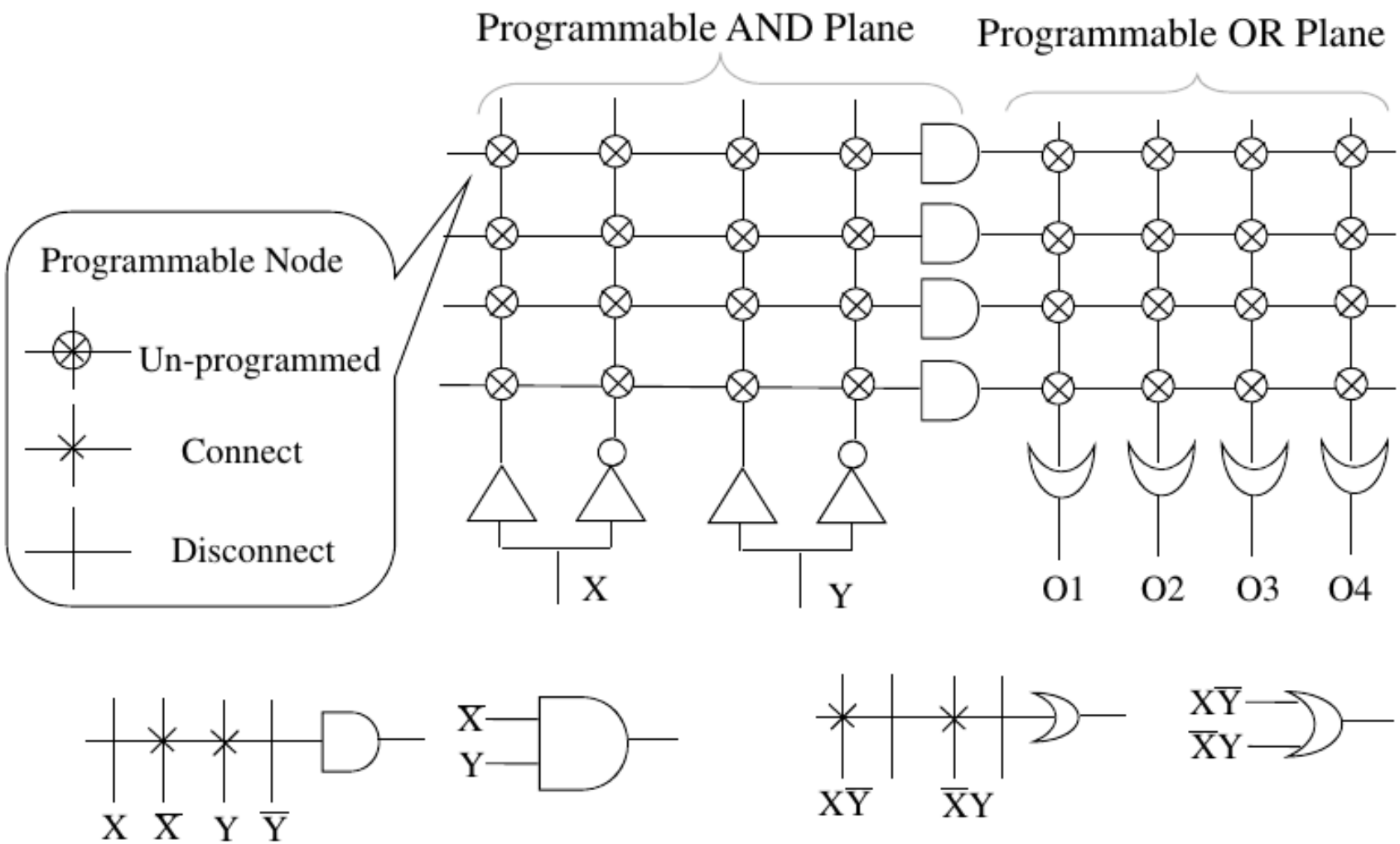
PLAs: Programmable Logic Arrays



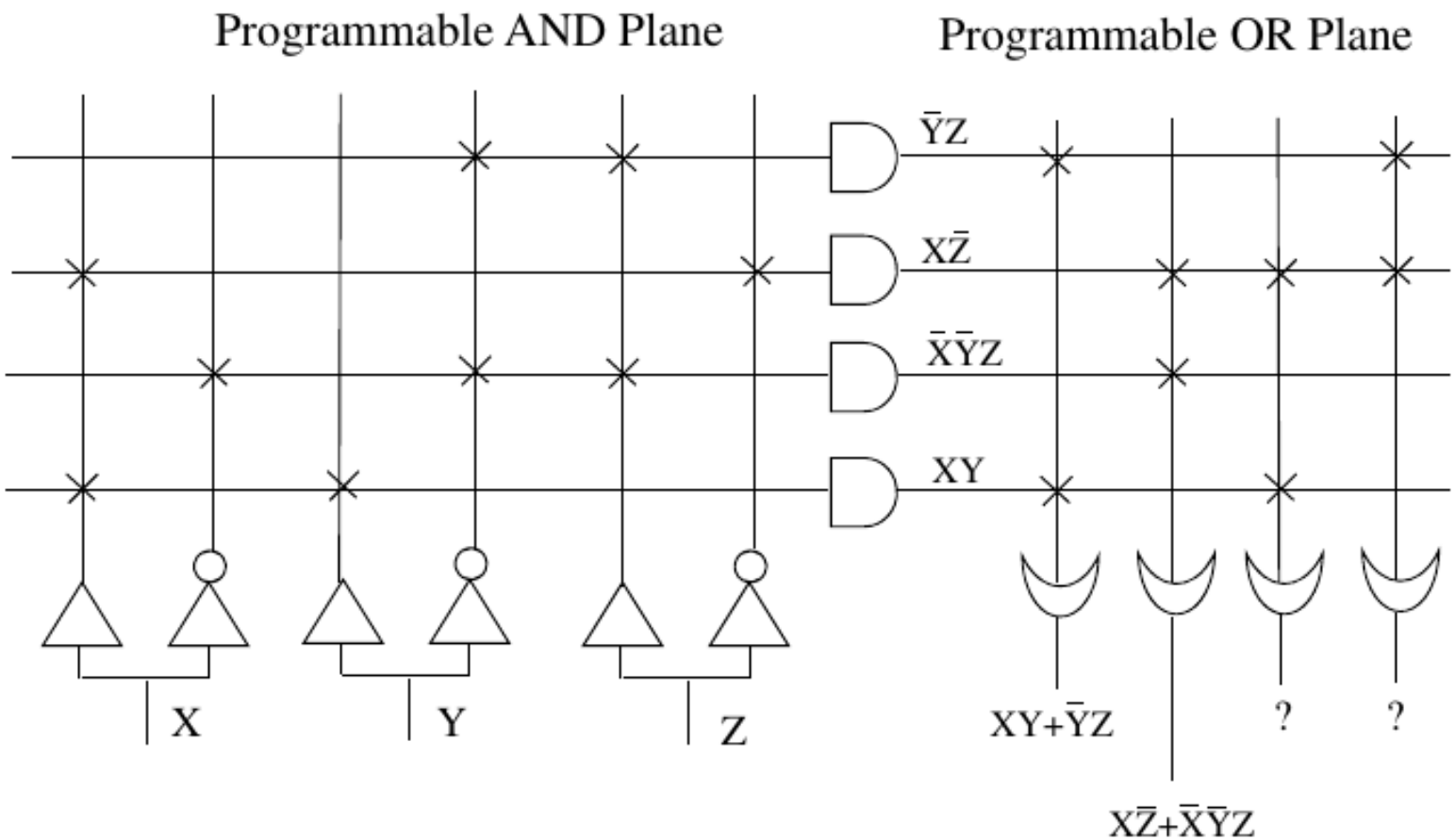


PLA Architecture

PLA


Programmable AND Plane
Programmable OR Plane
Programmable Node
Un-programmed
Connect
Disconnect
X
Y
O1
O2
O3
O4
X X̄ Y Ȳ
X
Y
XȲ
X̄Y
XȲ
X̄Y

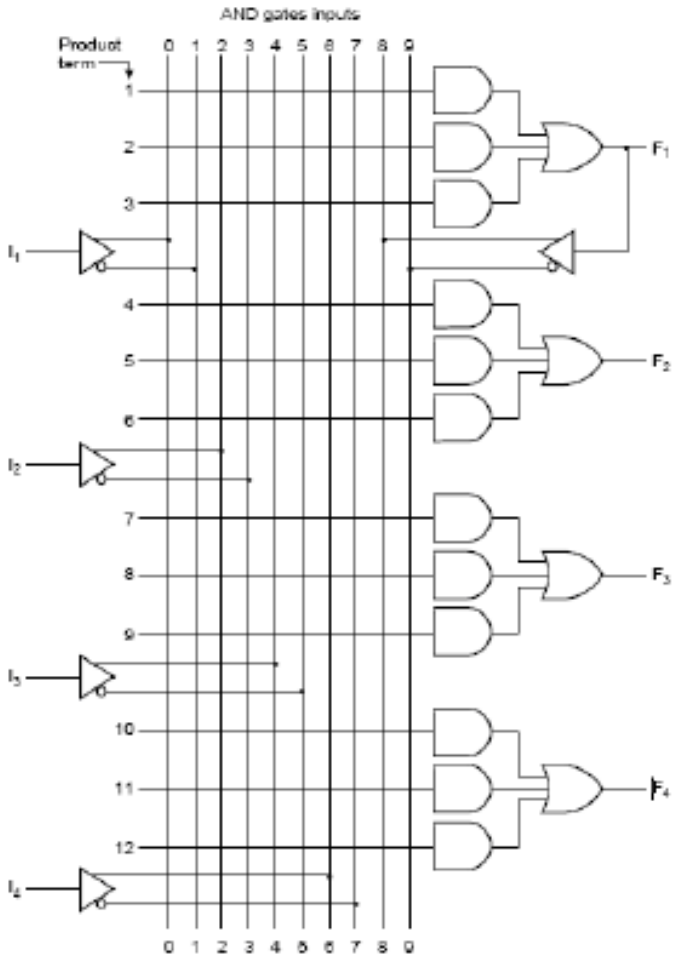
PLA

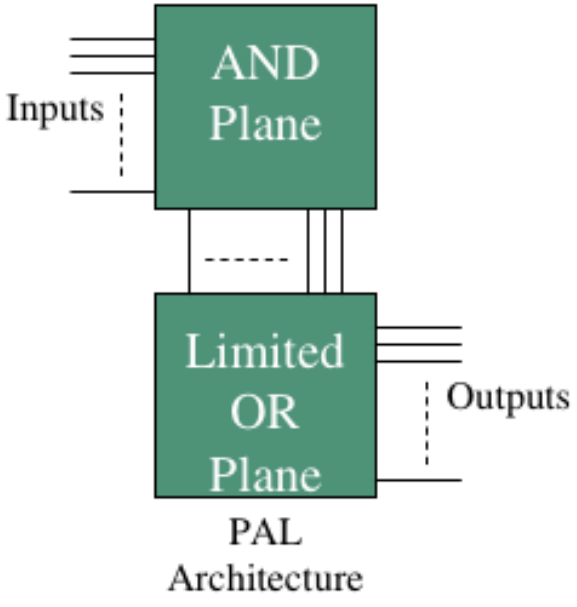

Programmable AND Plane
Programmable OR Plane
$\bar{Y}Z$
$X\bar{Z}$
$\bar{X}\bar{Y}Z$
XY
X
Y
Z
$XY+\bar{Y}Z$
?
?
$X\bar{Z}+\bar{X}\bar{Y}Z$

PALs: Programmable Array Logic

- A device similar to PLA.
- Introduced to overcome the weaknesses of PLAs at that time (programmable switches were hard to fabricate correctly and introduced significant propagation delays).
- A programmable AND array followed by a fixed OR array. Inverters at the inputs and outputs.
- HDLs to convert Boolean equations into connections

PALs: Programmable Array Logic





PAL Architecture

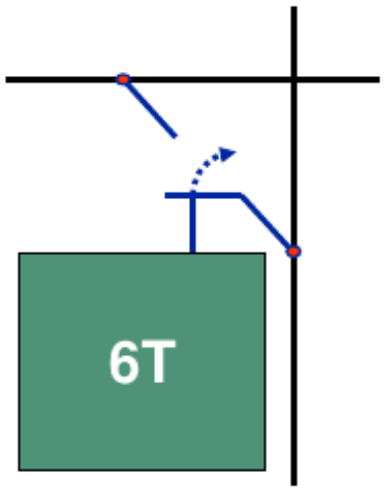
PLA *v.s.* PAL

- PLAs are more flexible than PALs since both AND & OR planes are programmable in PLAs.
- Because both AND & OR planes are programmable, PLAs are expensive to fabricate and have large propagation delay.
- By using fix OR gates, PALs are cheaper and faster than PLAs.
- Logic expanders increase the flexibilities of PALs, but result in significant propagation delay.
- PALs usually contain D flip-flops connected to the outputs of OR gates to implement sequential circuits.
- PLAs and PALs are usually referred to as SPLD.

FPGA Memory Types

T: transistor count

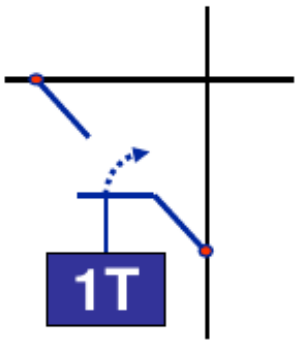
SRAM



Reprogrammable

Large Switch
expensive wires
Low Logic Utilization
typ 60%

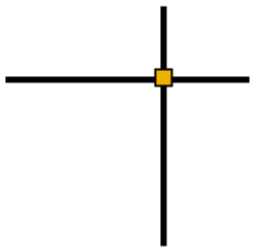
Flash



Best of Both Worlds
Reprogrammable
& Nonvolatile

Small Switch
cheap wires
High Logic Utilization
typ >85%

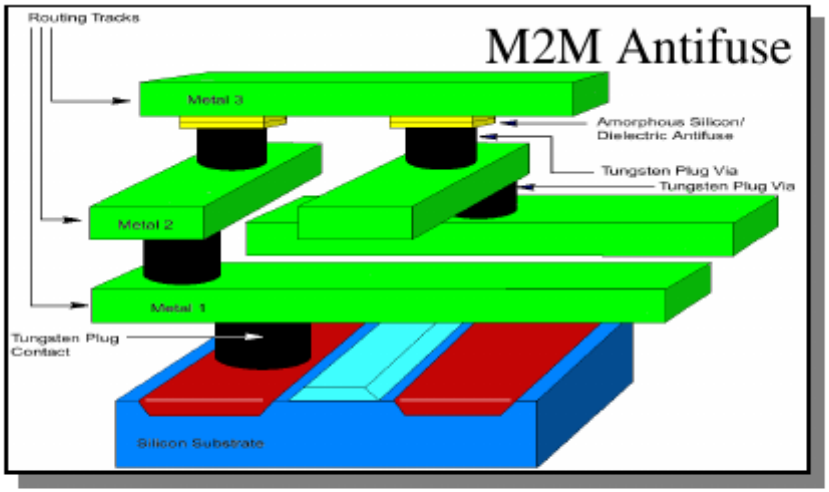
Anti-fuse

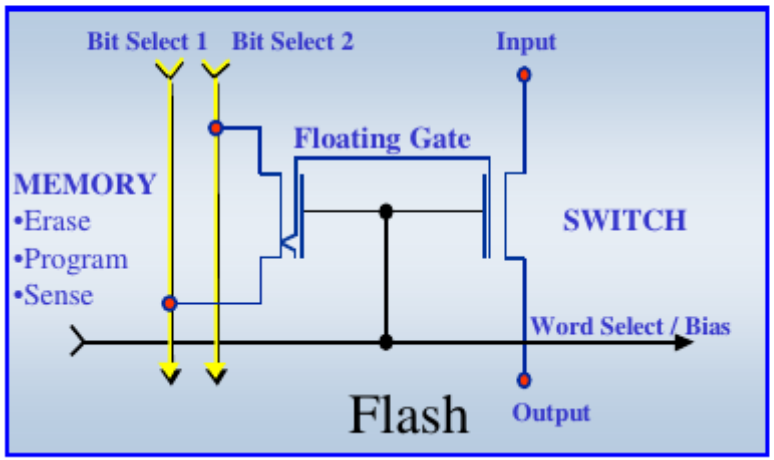


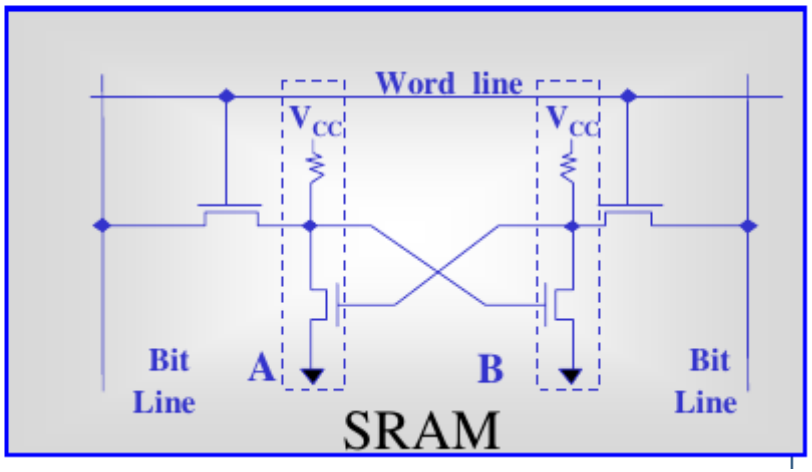
Nonvolatile

Smallest Switch
cheapest wires
Highest Logic Utilization
typ >90%

FPGA Memory Types

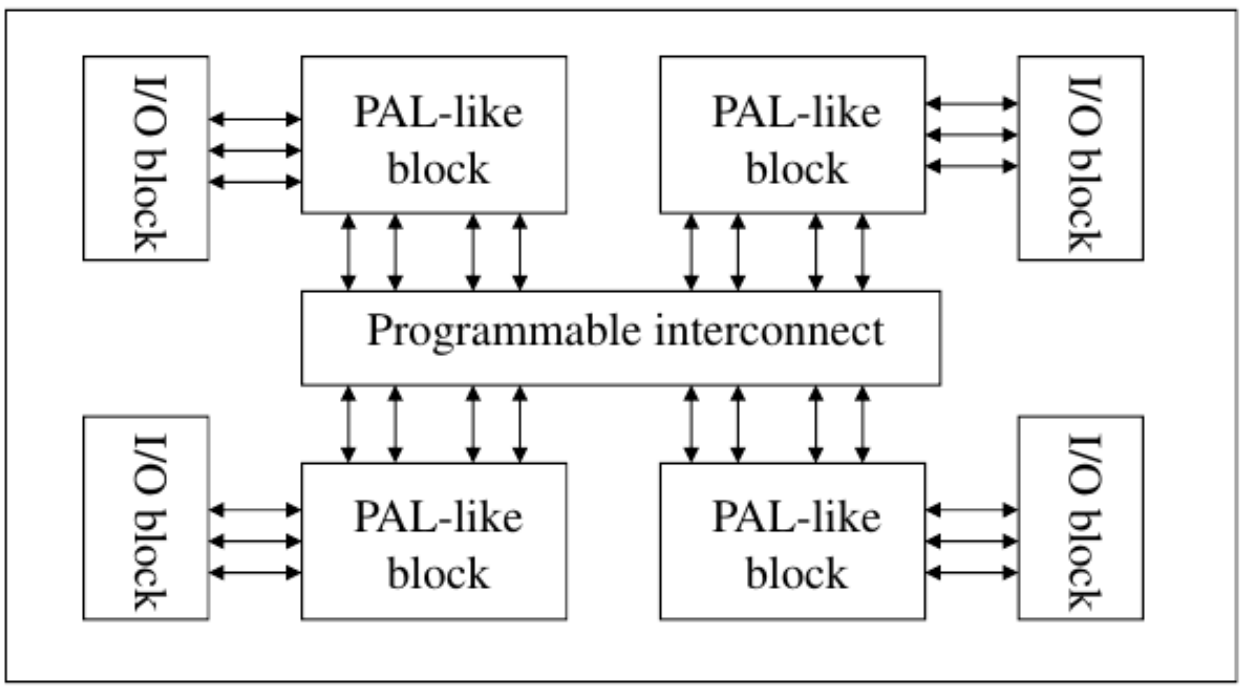




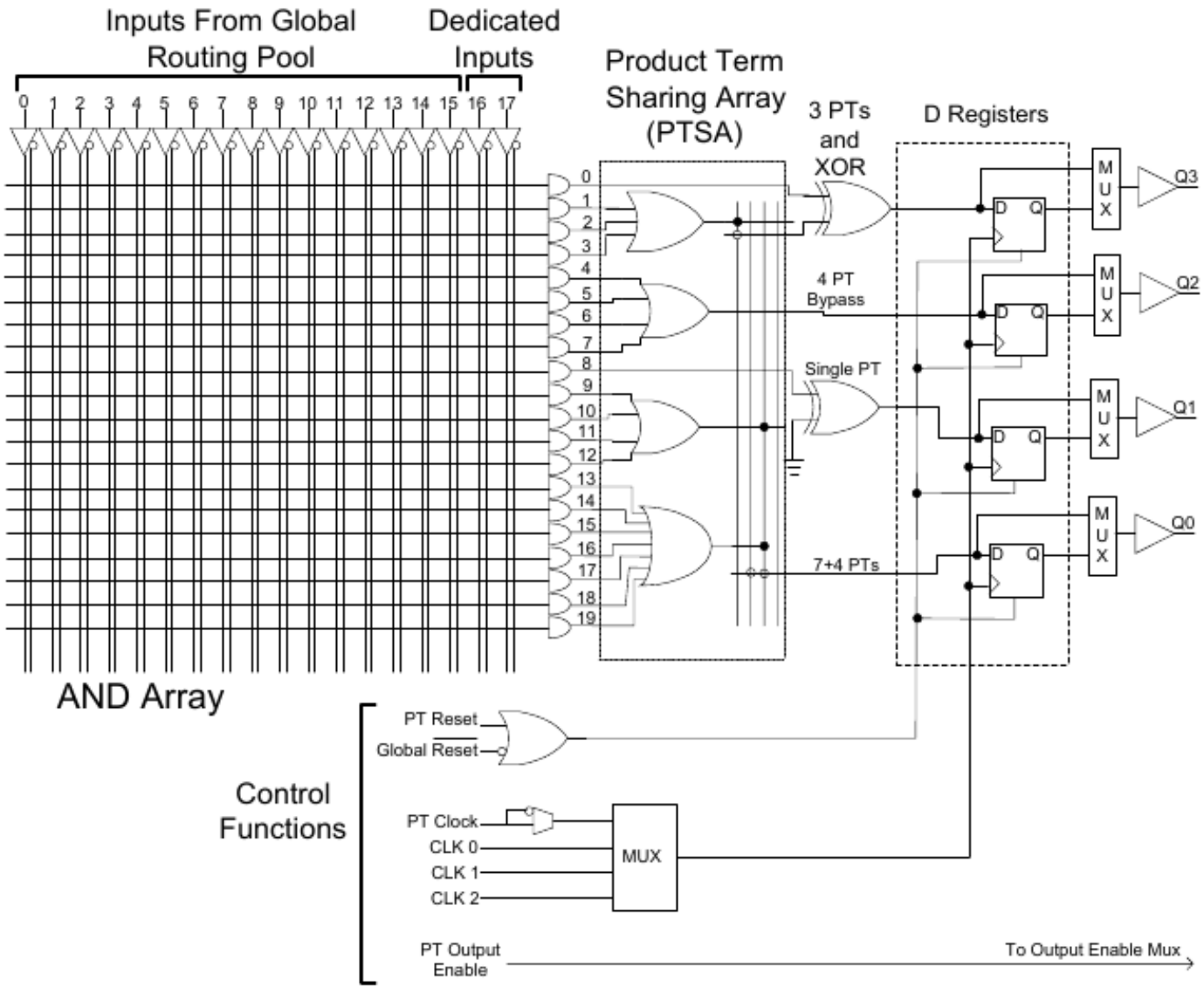


CPLD

- A CPLD comprises multiple PAL-like blocks on a single chip with programmable interconnect to connect the blocks.
- CPLD Architecture



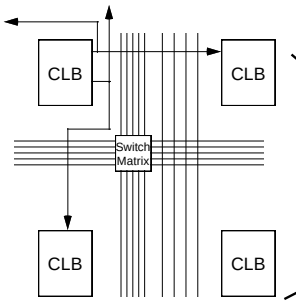
CPLD: Lattice Semiconductor ispLSI 2000/V



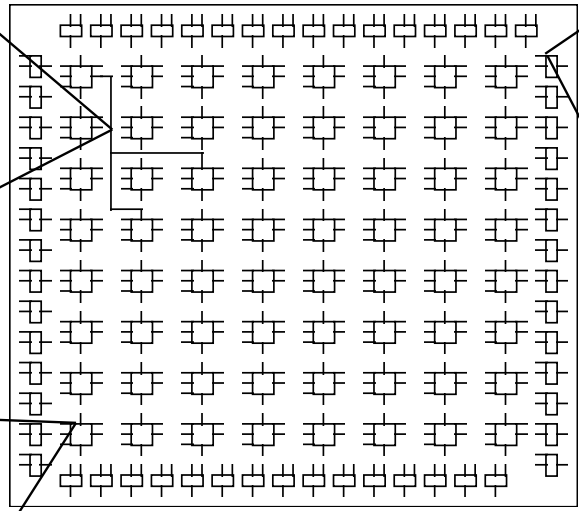
Xilinx FPGA Family

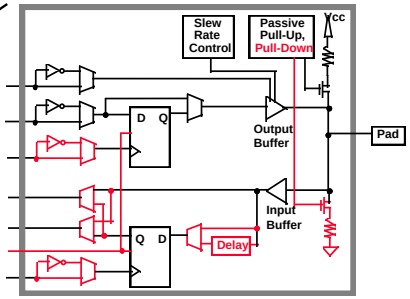
- Virtex Family
  - SRAM Based
  - Largest device has 1M gates
  - Configurable Logic Blocks (CLBs) have two 4-input LUTS, 2 DFFs
  - Four onboard Delay Locked Loops (DLLs) for clock synchronization
  - Dedicated RAM blocks (LUTs can also function as RAM).
  - Fast Carry Logic
- XC4000 Family
  - Previous version of Virtex
  - No DLLs, No dedicated RAM blocks

XC4000 Architecture

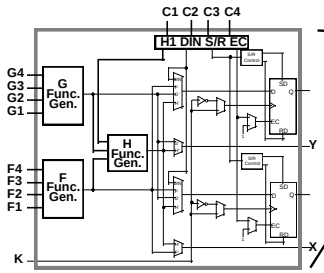


Programmable Interconnect





I/O Blocks (IOBs)



Configurable Logic Blocks (CLBs)

XC4000E/X Configurable Logic Blocks

- 2 Four-input function generators (Look Up Tables)

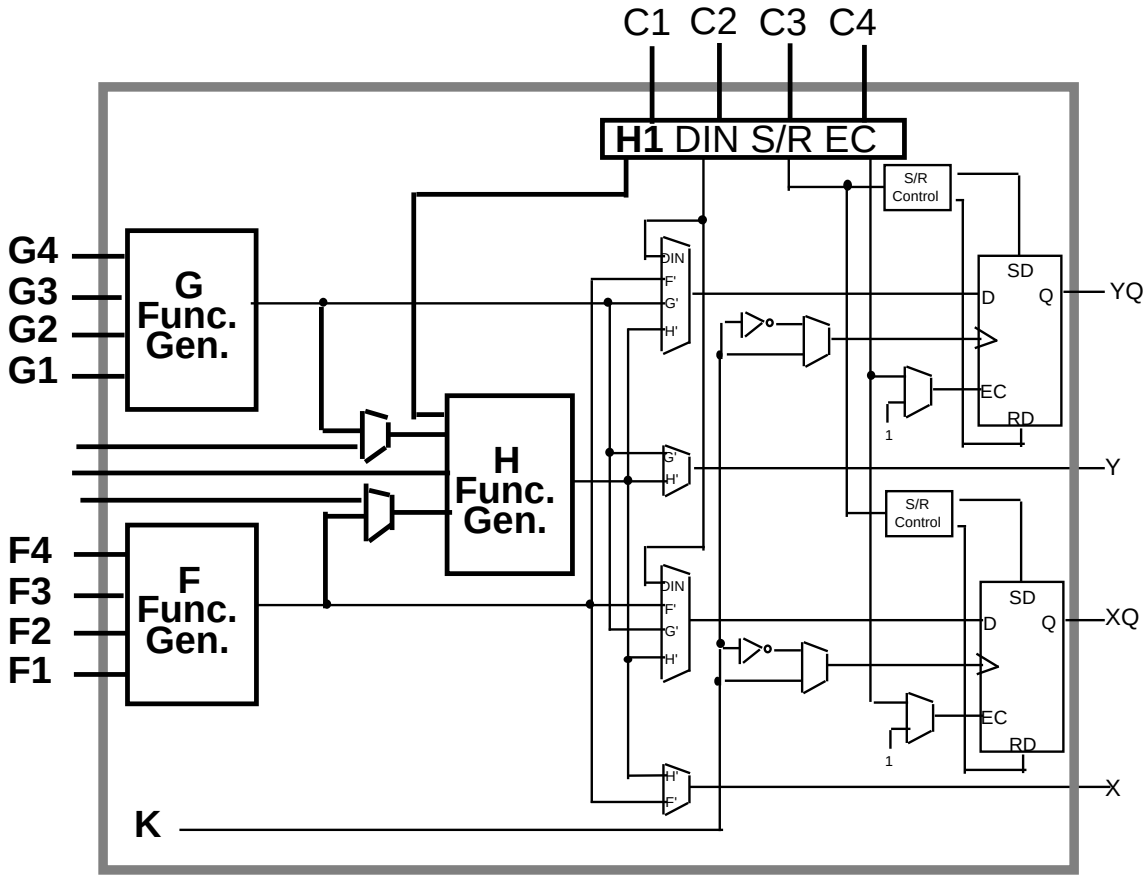
  - 16x1 RAM or Logic function

- 2 Registers

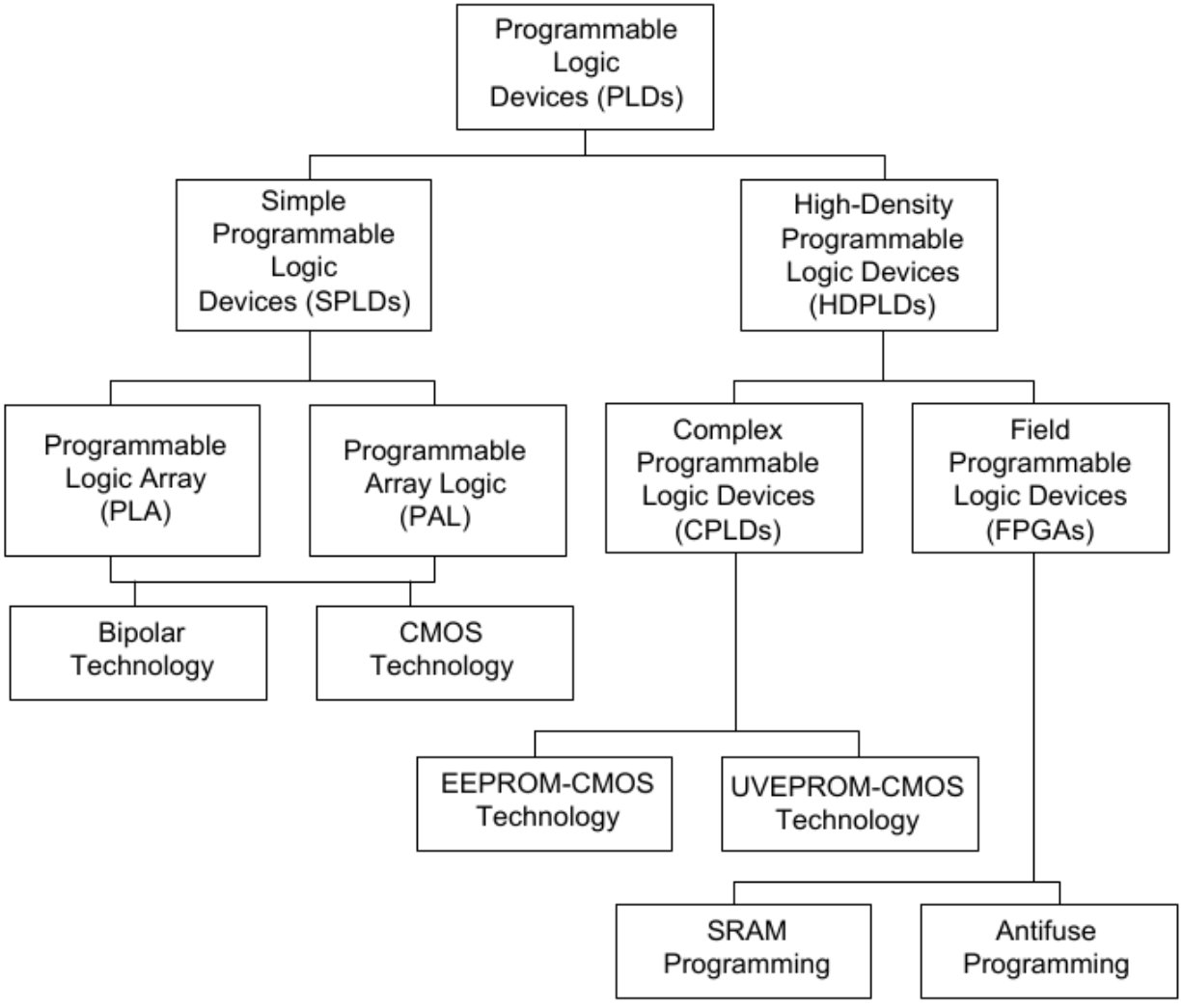
  - Each can be configured as Flip Flop or Latch

  - Independent clock polarity

  - Synchronous and asynchronous Set/Reset



Programmable Logic Devices

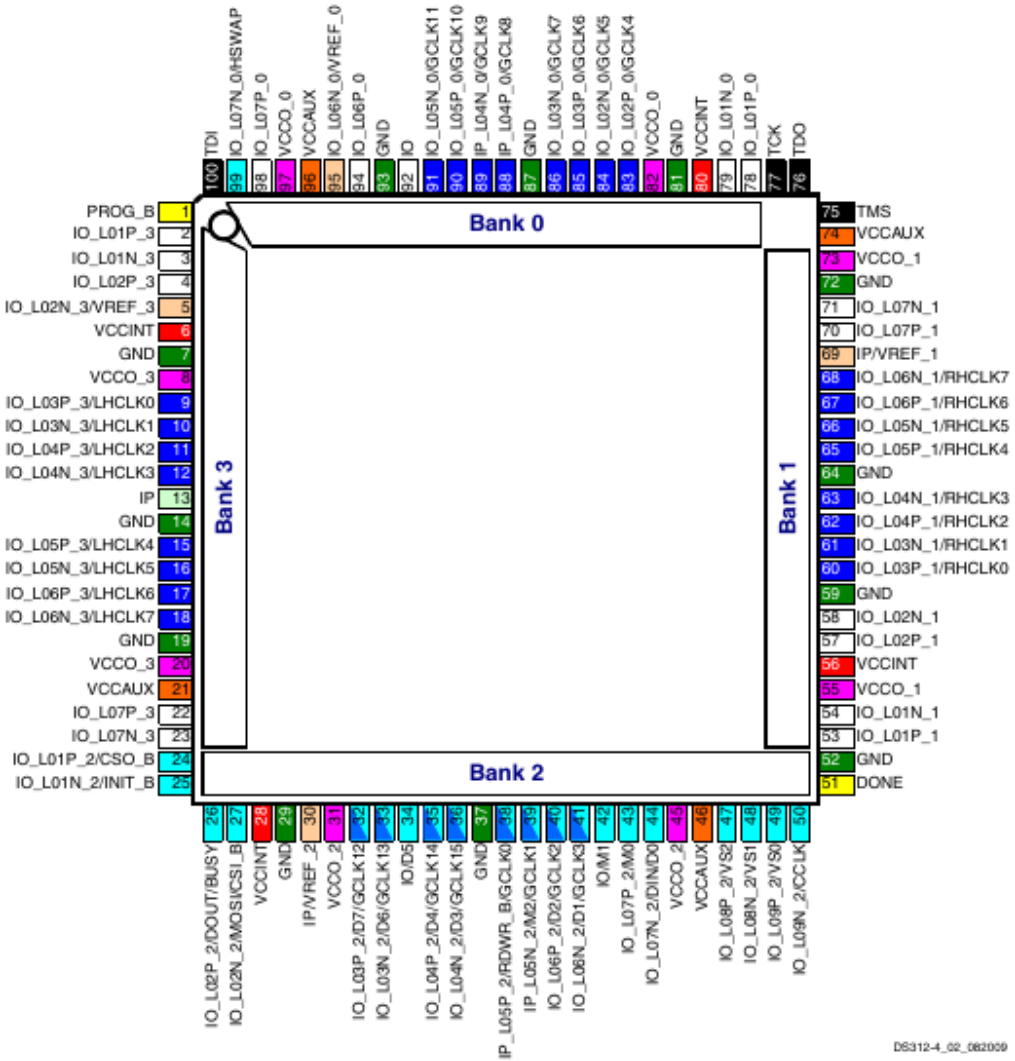


Xilinx Sparan 3A FPGA

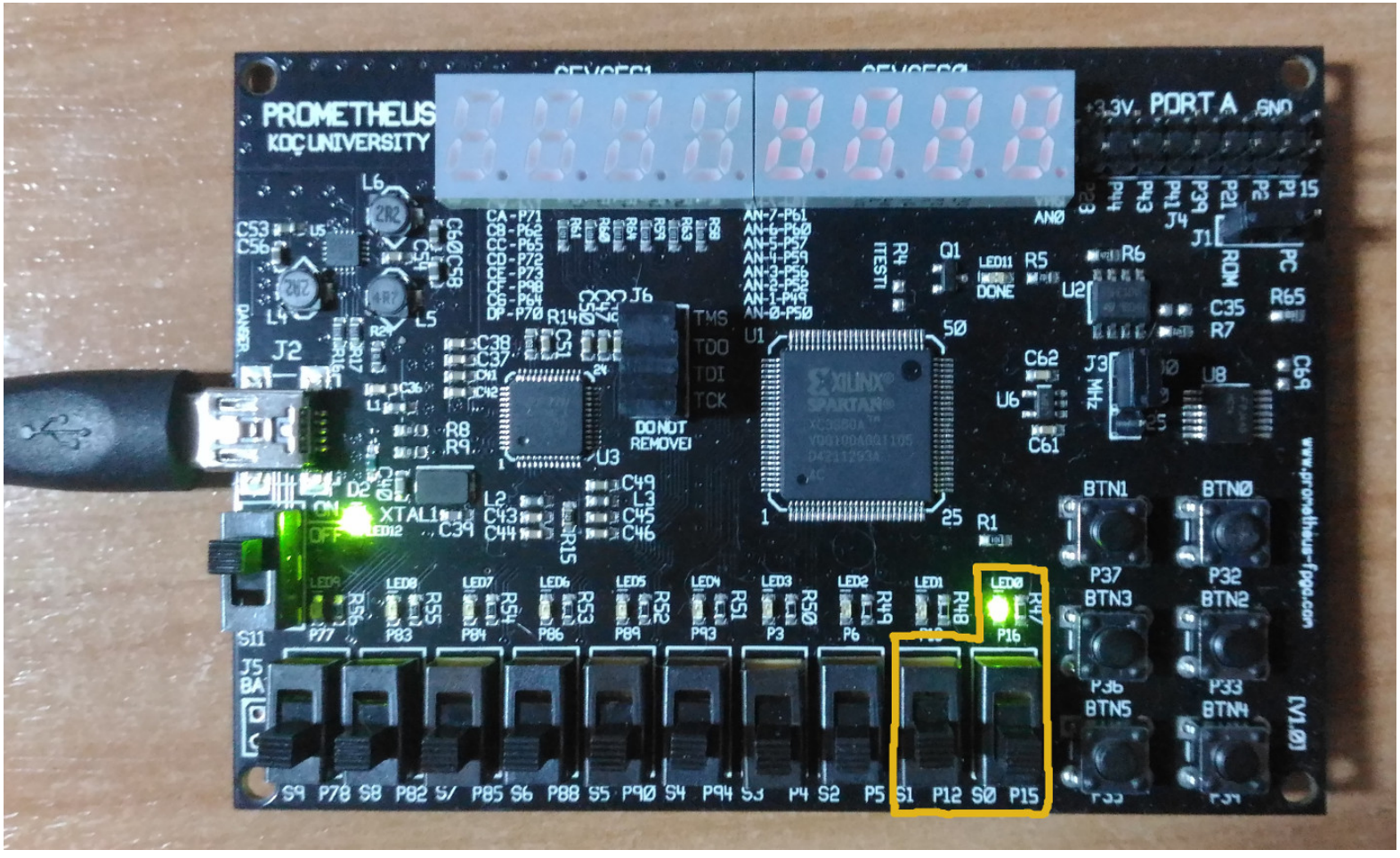
The FPGA chip looks like this:

Xilinx Sparan 3A FPGA

Pin-outs of the chip:



Prometheus Development Board



Hardware Description Languages

HDL Overview

- Hardware Description Languages
  - Used to model digital systems
  - Can model anything from a simple gate to a complete system
  - Support design hierarchy
  - Support Hardware Design Methodology
- Can model “real” hardware (synthesizable)

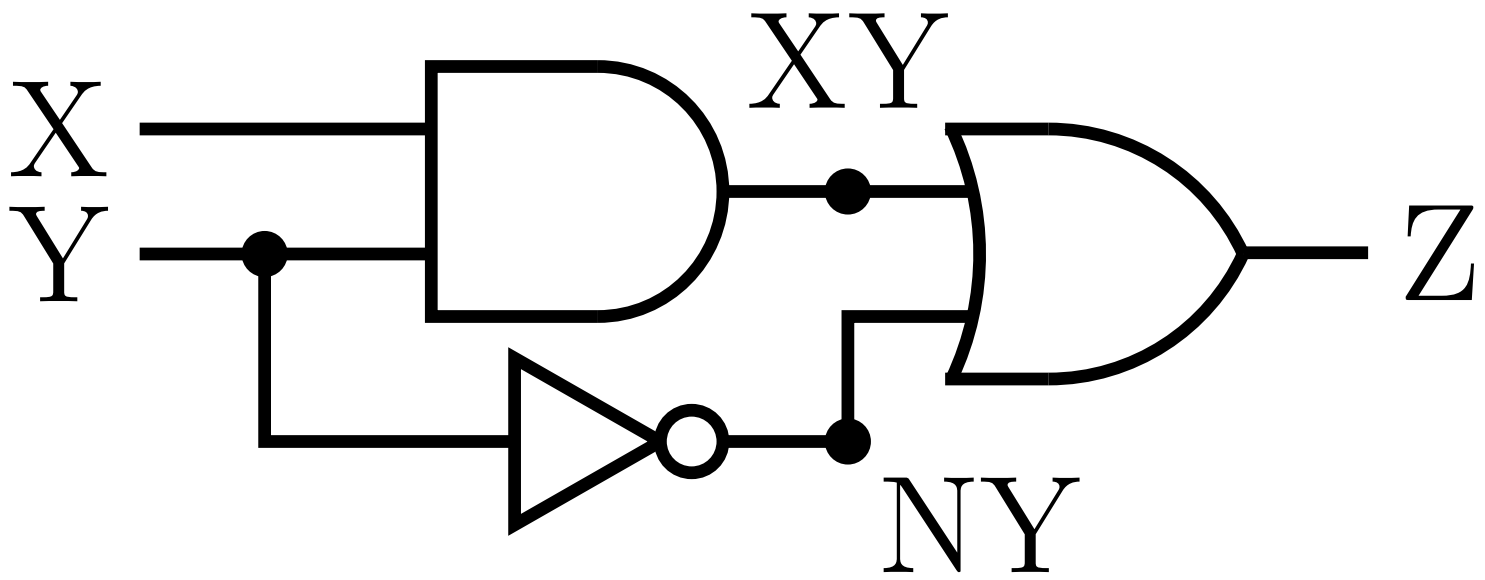
VHDL

- VHDL: VHSIC (Very High Speed Integrated Circuit) Hardware Description Language
- VHDL was originally developed at the behest of the U.S Department of Defense
- The IEEE Standard 1076 defines the VHDL
- The idea of being able to simulate the integrated circuits from the information in the documentation
- VHDL is **NOT a programming language**, it is a **hardware description language**

Why Use HDL?

Question:
How do we know that we have not made a mistake when we manually draw a schematic and connect components to implement a function?
Answer:
By describing the design in a high-level (=easy to understand) language, we can simulate our design before we manufacture it. This allows us to catch design errors, i.e., that the design does not work as we thought it would. Simulation guarantees that the design behaves as it should.

A Simple Logic Circuit



A Simple VHDL Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Lab1Code is
    Port ( X : in  STD_LOGIC;    -- Input signal
           Y : in  STD_LOGIC;    -- Input signal
           Z : out STD_LOGIC); -- Output signal
end Lab1Code;

architecture Behavioral of Lab1Code is
   signal XY : STD_LOGIC; -- Intermediate signal
   signal NY : STD_LOGIC; -- Intermediate signal
begin
   XY <= X and Y; -- Intermediate signals are defined here
   NY <= not Y;
   Z <= XY or NY; -- Output signal
end Behavioral;
```

A Simple VHDL Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Lab1Code is
    Port ( X : in  STD_LOGIC;    -- Input signal
           Y : in  STD_LOGIC;    -- Input signal
           Z : out STD_LOGIC); -- Output signal
end Lab1Code;

architecture Behavioral of Lab1Code is
    signal XY : STD_LOGIC; -- Intermediate signal
    signal NY : STD_LOGIC; -- Intermediate signal
begin
    XY <= X and Y; -- Intermediate signals are defined here
    NY <= not Y;
    Z <= XY or NY; -- Output signal
end Behavioral;
```

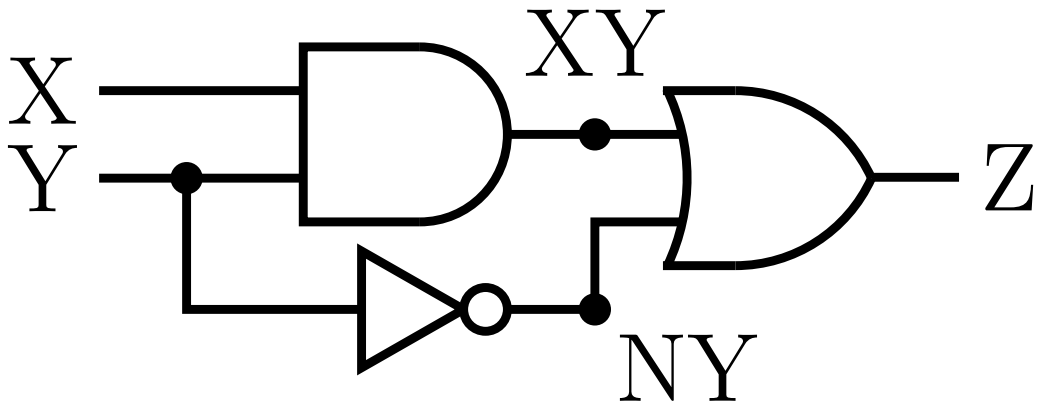
Libraries used

Input/Output definitions

Intermediate signals

Hardware description

A Simple Logic Circuit Description Using VHDL Code




```
XY <= X and Y;
NY <= not Y;
Z <= XY or NY;
```